AMENDMENT

In the Specification

Please amend the specification as follows:

Please replace the paragraph beginning on page 1 at line 15 with the following paragraph:

A primary problem with legacy busses is that they are difficult to configure. This was one of the motivations for developing the PCI bus, which introduced "plug and play" functionality. Plug and play functionality enables operating systems and other computer software and hardware to become apprised of a PCI ~~peripherals~~ peripheral's capabilities and characteristics. For example, on a first reboot an operating system may be able to determine that a PCI card that was installed prior to the reboot is a video card or modem with certain characteristics, and may further automatically configure the device, including identifying appropriate device drivers. This has enhanced usability of computers with PCI buses, especially when the computers are used by people with little or no technical background.

Please replace the paragraph beginning on page 2 at line 1 with the following paragraph:

While configuring PCI devices on a signal root bus is generally handled well by today's computers, it is anticipated that more powerful computers and servers will be introduced that support a variety of different interface and peripheral types through the use of multiple root busses. In some configurations, these root busses may comprise fundamentally different types of root busses. At present, the particular requirements of the chipset that control the each root bus are usually needed to configure the bus. In more particular, it is usually necessary to determine access ~~mechanism~~ mechanisms, resource constraints, I/O access mechanisms and/or parent-child relationships to configure the bus. With the introduction of the Intel 870 chipset, Infiniband bus protocol,

and IA-64, the process for controlling and configuration root busses will likely become even more complicated.

Please replace the paragraph beginning on page 5 at line 2 with the following paragraph: (Please note the text GRB (GUID of PPI for Root Bus) has not been changed)

The present invention provides a method for representing root busses and their subordinate bus configurations using an object oriented abstraction scheme that enables various system components to ~~communication~~ communicate with peripheral devices attached to the root busses and their subordinate busses without requiring specific knowledge of the access mechanisms of those devices. During the initialization process of a platform, a core dispatcher loads a PCI bus plug-in for each entity that can create a root bus. When the plug-in for an entity is loaded, it may produce a GUIDed object called a GRB (GUID of PPI for Root Bus) that provides an abstracted representation of the root buss's configuration and resources. The GRB includes a plurality of components including driver methods that may be used to enumerate the root bus corresponding to the GRB. Once the GRB is created, it is published to enable access to devices in the root bus's hierarchy.

Please replace the paragraph beginning on page 6 at line 14 with the following paragraph:

Busses between levels are enabled to communicate with one another through use of "bridges." The primary purpose of a "bridge" is to interface one bus protocol to another. The protocol includes the definition of the bus control signals lines, and data and address sizes. For example, a host/PCI bridge 0 is used to enable communication between host bus 12 and PCI bus 0. Under conventional terminology, a bridge is labeled to correspond to its subordinate bus, i.e., a bridge "n" will corresponding to a PCI Bus "n" or other type of Bus "n." When a bridge interfaces similar bus types, the bridge primarily limits the loading [[or]] on each bus. Instances of these types of bridges

are illustrated by the various PCI/PCI bridges in FIGURE 1. Bus configuration 10 also includes several PCI peripheral devices, including a modem 20, a sound card 22, and a network card 24. For clarity, many of the busses shown in bus configuration 10 are depicted as not being connected to any devices; it will be recognized that each of the busses may support one or more devices, and the principles of the invention may be applied to any of the busses, regardless of its configuration.

Please replace the paragraph beginning on page 8 at line 4 with the following paragraph:

With reference to FIGURE 3, a process for creating the GUIDed objects begins in a block 40 in which a core dispatcher loads plug-ins (i.e., software drivers) for entities that can create root busses. The core dispatcher comprises a core software component that is responsible for initializing and/or registering a plug-in. The entities that can create root busses typically may include chipsets that are used to control access to a root bus. For example, many PCI busses are controlled by an Intel ~~[please provide an Intel chipset for a PCI bus]~~ 82870 chipset. The loading of plug-ins generally may occur during the POST (power-on self-test) operation of the platform, or optionally during a similar platform initialization operation.

Please replace the paragraph beginning on page 9 at line 30 with the following paragraph:

Registration of each GRB and its associated methods comprises storing information in memory using an abstracted data structure comprising a handle that includes the GUIDs for the GRBs and a ~~pointers~~ pointer to ~~the~~ each GRB's memory location. With reference to FIGURE 4, this task is performed by looped process that is used during the registration of each root bus, as provided by respective start loop and end loop block 60 and 62. In a decision block 64 a determination is made to whether a handle has been allocated to store the registration information. During the first pass, a handle will not yet have been allocated, and so the answer to decision block 64 will be

no, and the logic will proceed to a block 66 in which a new handle will be created with the GUID for GRB of the first root bus that is evaluated. The logic then proceeds to a block 68 in which a ~~point~~ pointer to the GRB is attached opposite the GRB's GUID in the handle. At this point, the handle appears as a handle 70 depicted in FIGURE 5A, which includes a first GUID entry (RB0) that identifies the GRB, and a corresponding pointer (*RB0 GRB) to the GRB.

Please replace the paragraph beginning on page 13 at line 3 with the following paragraph:

[[The]] In a block 92, resources are allocated and set for the subordinate busses: